EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Adaptive routing for mobile ad hoc networks

Jeroen Hoebeke*, Ingrid Moerman and Piet Demeester

Abstract

Developing efficient routing protocols for mobile ad hoc networks remains a challenging task. It is well known that every protocol is capable of outperforming the others depending on the network context under which it is evaluated, since protocols only perform optimally under specific network contains due to their inability to adapt their behavior to the network context, which both varies in time and place. This article builds upon this observation to motivate and propose an adaptive multi-mode routing framework that has multiple compatible modes of operation. Based on this framework, an adaptive protocol has been implemented with the novel feature that individual nodes can adapt their mode of operation at any moment, while an overall consistent state of the routing tables is maintained. Through simulation, the correct behavior of the protocol during mode switches is demonstrated and it is shown that the protocol is capable of minimally offering the performance of either proactive or reactive routing, Its capabilities to dynamically switch, when intelligently applied, allow outperforming these protocols. This is illustrated for one specific application scenario where network conditions are dynamically monitored. Further, the article discusses some challenges encountered during the design and, since no monitoring solution has been developed, identifies existing solutions for monitoring and dissemination of network context are identified, offering directions for further research.

Keywords: wireless ad hoc networks, routing, adaptive, simulation, multi-mode

1. Introduction

During the past decade, advances in mobile computing and wireless communication technologies have led to wireless networks offering connectivity to mobile users. One important type of mobile wireless networks is an ad hoc network. Opposed to infrastructured wireless networks, where each user directly communicates with an access point or base station, a mobile ad hoc network or MANET does not rely on a fixed infrastructure for its operation. This type of network is suited for use in situations where a fixed infrastructure is not available, not trusted, too expensive, or unreliable. Possible scenarios include, but are not limited to: emergency and rescue operations, conference or campus setting, temporary headquarters or military operations. The network is an autonomous system that consists of mobile nodes communicating with each other over wireless links [1]. Nodes that lie within each other's send range can communicate directly. In order to enable connections between nodes that are not directly within each other's send range, intermediate nodes act as routers that forward packets to the destination. Hence, routing is one of the primary functions; each node has to perform in order to have a fully functional network.

Developing efficient routing protocols has always been a challenging task because of the specific characteristics of a MANET environment [2]. First, since nodes are free to move arbitrarily, the network topology may change randomly and rapidly at unpredicted times and an efficient routing protocol should be able to react appropriately to these topology changes. Second, the available bandwidth is limited because it is shared by multiple nodes and can also vary due to fading, noise, and interference. As a consequence, the protocol's amount of control information should be limited. Third, the nodes that form the network can be very heterogeneous, ranging from small battery-powered devices with limited processing capacity to full-fledged computers possibly connected to the power net and with ample processing power. Finally, network sizes can range from tens up to thousands of nodes.

Therefore, the development of efficient ad hoc routing protocols, one of the fundamental requirements to have an operational ad hoc network, has always received a lot

* Correspondence: jeroen.hoebeke@intec.ugent.be
Department of Information Technology (INTEC), Ghent University - IBBT, Gaston Crommenlaan 8 Bus 201, 9050 Ghent, Belgium

of attention in the research community and their comparison in function of the network conditions remains a research topic. This has lead to a wide range of ad hoc routing protocols [3,4]. Nevertheless, it is still a well-known problem that existing protocols are only suboptimal. Every protocol is capable of outperforming the others depending on the network context under which it is evaluated. No perfect routing protocol exists, a limitation that is caused by their inability to adapt their behavior to the context of the network, which both varies in time and place.

The work presented in this article builds upon this observation to motivate and investigate the feasibility of an adaptive multi-mode routing protocol framework that has multiple compatible modes of operation, where each mode can be designed to operate as efficiently as possible in a given networking context. Based on this framework, an adaptive multi-mode routing protocol is implemented offering three different modes of operations. Nodes can adapt their mode of operation at any moment, while a consistent state of the routing tables is maintained and without impacting on-going connections. To the best of authors' knowledge, this goes beyond current research into adaptive ad hoc routing protocols, since no adaptive protocol described in literature has the ability to let individual nodes change their routing technique while an overall consistent state of the routing tables is maintained.

In Section 2, we motivate the need for adaptive ad hoc routing and give an overview of related study. Section 3 gives an overview of the proposed framework. In Section 4, we describe in detail the translation of the generic framework into an actual implementation in a network simulator, resulting in a protocol offering three fully compatible modes of operations. This section also clearly describes what steps need to be taken and what complexity is involved in order to achieve real mode compatibility. In Section 5, we demonstrate through simulations that individual nodes can independently switch mode and that nodes with different nodes can move around, while the protocol keeps it ability to establish routes and forward traffic because of the realized mode compatibility. At the same time, the performance is compared with traditional proactive and reactive ad hoc routing. Finally in Section 6, we illustrate with a simple example where network conditions are monitored and the protocol adapts to changes in the network context that the implemented protocol is capable to outperform proactive and reactive protocols; thanks to its capabilities to dynamically switch routing strategy. Since no real monitoring solution has been developed, we also identify existing solutions for monitoring and dissemination of network context. Finally, some other potential application domains of the protocol are mentioned. We conclude this study in Section 7.

2. The need for adaptive routing

Essentially, ad hoc routing protocols can be categorized in the following two classes depending on the way they find routes: proactive routing protocols and reactive routing protocols. Proactive routing protocols or table-driven routing protocols attempt to have at all times an up-to-date route from each node to every possible destination. This requires the continuous propagation of control information throughout the entire network in order to keep the routing tables up-to-date and to maintain a consistent view of the network topology. Proactive routing protocols differ in the number and type of routing tables and the way in which topology changes are broadcasted. On the other hand, reactive protocols or on-demand routing protocols only set up routes when needed. When a node requires a route to a destination, the node initiates a route discovery procedure by broadcasting a route request within the network. Once the route is established, a route maintenance procedure is responsible for keeping the active routes up-to-date as long as needed. When a link break occurs, a route repair procedure may be started. Different reactive protocols have different strategies to deal with route maintenance and route repair.

Most other types of routing protocols can be seen as variants of the 'all routes at all times' or 'one route when needed' paradigm of the proactive and reactive techniques, respectively [5]. Hybrid routing protocols try to combine proactive and reactive techniques in order to reduce protocol overhead. Position-based routing protocols use geographical information to optimize the routing process. Finally, hierarchical protocols, such as clustering protocols, introduce a hierarchy in the network in order to reduce the overhead and to improve the scalability. In the remainder of this article, we focus on the fundamental proactive and reactive techniques.

Basically, both approaches rely on the propagation of control messages throughout the entire network in order to establish routes, but the way in which the broadcasting of control messages is applied differs completely. As a consequence, their performance will be different, with one technique outperforming the other depending on the network conditions, proven by many simulation studies that have been conducted in order to evaluate the performance of proactive and reactive routing protocols [6].

Ideally, devices should choose the optimal routing technique depending on the type of ad hoc network they participate in and the current network conditions in this network. We believe this is not feasible by letting

a single protocol adapt to the network conditions (e.g., by changing parameter values), but requires the support of multiple different routing strategies. Hybrid routing protocols such as the Zone Routing Protocol [7], Fisheye State Routing [8], SHARP [9], and the Independent Zone Routing (IZR) framework [10] are already a first step for the development of routing protocols that combine multiple routing techniques. However, they do not obtain the degree of adaptivity we envision, since the used routing strategy is bound to specific zones. Within the IETF MANET Working Group, work has targeted the development of both a stand-alone reactive (DYMO [11]) and proactive (OLSRv2 [12]) routing protocol, only striving to introduce a common packet format (PacketBB [13]) and neighborhood discovery protocol (NHDP–[14]) for ad hoc networks. In [15], the AROD scheme is presented, which is a seamless integration of several existing schemes and is claimed to be the first routing scheme that is adaptive to network density as well as to mobility patterns. However, this protocol focuses specifically on delay tolerant networks where a contemporary path between a source and a destination might not exist, and the delivery of messages must utilize node mobility. The scheme therefore combines both multi-hop and multi-copy forwarding and differs from the work we present here, which does not make any assumptions on the type of ad hoc network or context.

Another alternative is to implement different protocols in every mode and switch protocol when the need occurs. This has been proposed in [16], where it is possible to switch routing module, next to tuning routing parameters or routing metrics. The authors state that this requires resetting and reconfiguring the routing service, which may cause service discontinuity. On top of this, it requires a consensus protocol on when and how to make the change. This has several limitations, which we want to avoid by allowing routing strategy switching by individual nodes (or group of nodes) without any service discontinuity.

To this end, we propose the development of an adaptive multi-mode routing protocol that has multiple compatible modes of operation (e.g., proactive, reactive, flooding, or variants), where each mode is designed to operate as efficiently as possible in a given networking context. Simulation studies, analytical studies, and models can be used to determine the optimal network conditions of the different modes. The main issue in the development of such a framework is that nodes need to be capable of monitoring and estimating the network conditions in their environment with as little overhead as possible. Based on these, predictions nodes can adapt their mode of operation to the networking context and perform the best possible routing. This has to be achieved without any service discontinuity or performance degradation. We see several advantages when adaptive routing could be introduced within an ad hoc network:

- *Improved efficiency by adaptation*: by its capability to adapt to the network, the routing protocol can provide better routing in networks with varying conditions. As mobile ad hoc networks are intrinsically characterized by a very dynamic nature, this is certainly a big advantage opposed to existing routing protocols that are not aware of the network context.
- *Compatibility*: when the modes are developed with built-in compatibility in mind, different modes in different parts of the network can coexist and local adaptations can be made.
- *User friendliness*: devices can participate seamlessly in different types of ad hoc networks without the need to manually switch to another protocol, because the protocol will adapt itself to the current network conditions.
- *Future proof*: the use of different modular modes could ease the future development of the protocol. Existing modes can be extended or enhanced or new modes can be added without the need to completely change or rethink the protocol design.

In the following section, we will present the general framework of the adaptive multi-mode routing protocol we propose.

3. Adaptive multi-mode framework

The framework shown in Figure 1 consists of two main components, namely a monitoring agent and the actual routing protocol, and a number of secondary components such as the neighbor discovery module and connection info module, which we will now discuss in more detail.

In order to enable connectivity in an ad hoc network through the exchange of routing information, an initial requirement is the ability to detect new links and link breaks. For instance, a proactive routing protocol needs to know about any new link and link break before routing update messages can be propagated. A reactive routing protocol can do without new link detection, but requires mechanisms to detect the failure of a link. To this end, an essential component of the routing protocol framework is the neighbor discovery module. This module is responsible for the detection of new links and link breaks and for relaying this information to the routing protocol. The information is stored in a neighbor table, which can be consulted by the routing protocol and monitoring agent. The detection of new links and link breaks can happen through the exchange of beacon messages. In addition, this exchange can take place at layer 3 or at layer 2 (as part of the radio interface implementation).

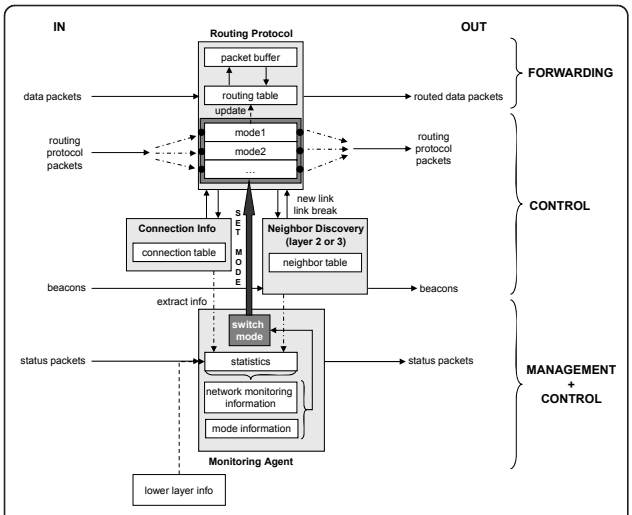

Figure 1 Adaptive multi-mode routing framework.

Next, there is the actual routing protocol. The functionalities of the routing protocol can be divided into forwarding and control. Incoming data packets are processed by the routing table and forwarded to the next hop based on the content of this table. This is the forwarding functionality. In order to fill in this routing table or forwarding table, propagation of routing protocol messages is required, which fulfills the control functionality of the routing protocol. As our novel protocol is a multi-mode protocol, control functionality is present for each available mode (as part of the mode module). When a routing protocol packet arrives, a mode parser determines the mode of the protocol packet and the packet is relayed to the appropriate mode control component. According to the content of the protocol packet, the mode component takes the appropriate action (e.g., a reactive mode will relay a route request or answer with a route reply) and, if necessary, updates the main routing table. This table contains all the valid route entries, possible coming from different modes. The different modes can also use the information in the neighbor table in order to improve their efficiency. Packets that cannot be routed immediately can temporarily be stored in the packet buffer. At each moment, only one mode is chosen as the active mode (as determined by the switch mode component), but protocol packets from nodes in another mode can also be received.

The monitoring agent is responsible for collecting information about the network conditions in the environment of the node. This is done in two ways. First, local statistics from the network layer or other layers are collected in the statistics component of the monitoring agent. These statistics can include, but are not limited to: the number of data packets routed, signal strength of the received packets, number of packets dropped due to congestion... Second, non-local statistics can be collected through the periodic propagation of status packets to the neighboring nodes. These status packets can contain statistics and network monitoring information collected by the sender, such as the observed network load and the mode of operation the node is currently in. In this way, statistics information can propagate gradually throughout a part of the network. Consequently, nodes are provided with information from their immediate environment.

Periodically, the network monitoring information component processes the collected statistics. This component is responsible for extracting useful information about the networking context such as the network load or mobility. Based on the simulation studies or analytical models, the mode information component must have knowledge under which network conditions each of the available modes their performance is optimal. This information, together with the information provided by the network monitoring information component, is used by the switch mode component to decide whether or not the node should switch to a more efficient mode of operation. If the node has to switch to another mode of operation, the routing protocol is informed. As such, the monitoring agent provides both control (exchange/collection of statistics) and management functionality (determining and setting the mode). Further, every node in the network will determine its most optimal mode at a given moment according to the network conditions. As we require the different modes in the network to be compatible (which will be explained later in more detail) with each other, different modes can be active in different parts of the network and nodes can independently decide to switch to another mode (although often neighboring nodes will experience more or less the same network conditions).

Finally, the framework also has a connection table that keeps track of requested and/or established connections. As such, the connection table provides valuable information on the traffic patterns in the network, which can be used to determine if a mode switch is necessary. In addition, this information can prove valuable for the routing protocol during a mode switch. Information about the requested and established connections is provided by the routing protocol component.

It is important to note that in theory each different mode should be implemented as a separate pluggable building block. As such, it is possible that a node does not implement all available or existing modes (e.g., due to memory or processing constraints) or dynamically plugs in a new mode when required. Dynamically, adding a new mode will involve the installation of a new mode module in the routing protocol component and the addition of rules that determine which actions need

to be taken upon a mode switch of this node or another node to or from this new mode.

4. Implementation

In this section, we will describe in detail the translation of the generic framework into an implementation in the network simulator GloMoSim [17] and the feasibility of implementing an adaptive protocol according to the proposed framework. We have chosen to use a simulator, the most commonly used design methodology when evaluating ad hoc networks, since this offers most flexibility in the evaluation of the correctness and performance of protocols for a wide range of network setups. Moreover, simulation results can easily be reproduced, facilitating development.

First, we will introduce the different modes that are implemented and argue why they are relevant and how they will interact. Next, we will describe in detail the protocol including message types, data structures, and protocol charts. This also includes a detailed explanation on how mode compatibility can be assured, how switches between the different modes are handled, and how new links and link breaks are dealt with.

4.1 Selection of the implemented modes and their interaction

For the adaptive multi-mode routing protocol, we have chosen to implement both a reactive and a proactive mode. The reactive mode is based on the basic principles used by AODV [18] and DYMO [11]. It uses a RREQ-RREP cycle to establish routes. The proactive modes use distance vectors for its operation (as in WRP [19]). We have chosen to use distance vectors and not a link-state protocol such as OLSRv2 [12] because we want to allow individual nodes to propagate their reachability (see further). This proactive mode can be further refined depending on the number of destinations for which the routing information is propagated proactively (ranging from a single destination to all destinations in the network) and the scope of propagation (ranging from only the neighbors to the entire network). Based on these considerations, we have chosen to use the following three modes:

• *MODE 1*: nodes in MODE 1 do not proactively announce their location through the propagation of distance vectors. If another node wants to communicate with this node, it has to establish a route reactively. As a basic requirement, all nodes must support MODE 1 (alternatively, the basis requirement could be that flooding must be supported).

• *MODE 2*: a group of interconnected nodes in MODE 2 will keep an up-to-date route to each other through the propagation of distance vectors. The propagation of these distance vectors is limited to nodes that are in MODE 2 or MODE 3 and support MODE 2, i.e., when a node in MODE 1 receives such an update it will not forward, but discard it. This mode actually creates clusters of nodes that proactively maintain routes to each other.

• *MODE 3*: nodes in MODE 3 will proactively announce their location through the propagation of distance vectors. In this case, all nodes that support MODE 3 (independent of the mode they currently use) are allowed to further propagate this information, which makes this mode different from MODE 2.

For the proactive modes (i.e., MODEs 2 and 3), the scope of propagation (i.e., locality) can be varied based on time-to-live values. When a node in MODE 2 or MODE 3 propagates a distance vector that announces its destination, a TTL value is included that determines how far this information may propagate in the network. Nodes that receive a MODE 2 or MODE 3 routing update with a TTL equal to 0, will not further propagate this information. From the basic operation of WRP, on which the proactive modes are based, one can easily derive that the following needs to hold (in order always to have a path to the predecessor of a destination, which is needed to avoid routing loops) for all nodes that support MODE 2: TTL ≥ MAX(TTL of all MODE 2 neighbors) - 1. Within one specific mode, further fine-tuning to the network conditions could be possible. For instance, the frequency of the periodic exchange of distance vectors can vary depending on the measured mobility. Another example could be the use of path accumulation, expanding ring search, or caching in the reactive mode.

At each moment, every node chooses its optimal mode. This mode determines how the node will propagate its information in the network and how the interaction with other nodes will be. Therefore, a node can have routing table entries for different modes in its routing table (depending on the modes of the other nodes in the network and the modes the node supports). For instance, a node in MODE 2 (and supporting MODEs 2 and 3) will have a MODE 2/MODE 3 entry in its routing table for a neighboring MODE 2/MODE 3 node. If it wants to communicate with a node that is not known proactively, this will create a MODE 1 routing entry in its routing table. As a consequence, the content of a node's routing table will depend on its mode, its supported modes and the interaction with the modes chosen by the other nodes. Figure 2 and Table 1 illustrate these mode interactions and their impact on the content of the routing tables.

4.2 Message types

The following message types are used by our adaptive protocol.

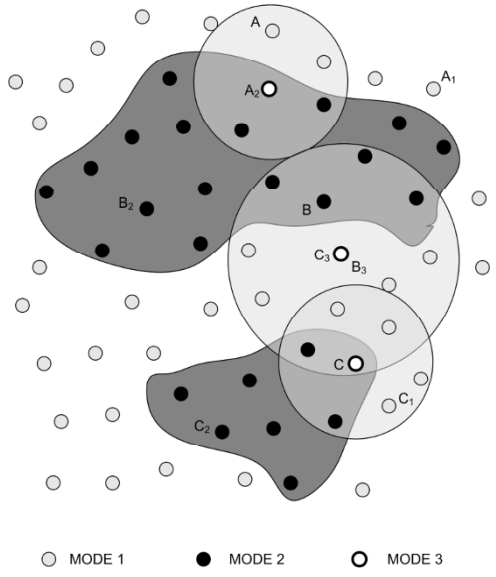

Figure 2 Overview of mode interactions and their impact on the routing table.

HELLO messages: hello messages are exchanged between neighboring nodes at periodic times. These messages are used to detect new links and link breaks, to inform neighboring nodes about the mode the sending node is currently in and about the modes supported by this node and optionally, to distribute information about the network conditions.

MODE 1 messages: MODE 1 is a reactive mode based on AODV [18] and implements in a similar way the well-known route request, route reply, and route error messages for its operation.

MODEs 2 and 3 messages: both proactive modes are based on WRP [19], which uses distance vectors. Similar messages and data structures have been used with some modifications to support both MODEs 2 and 3 and to limit the propagation based on time-to-live values. The format of a distance vector update entry is the following: (*destination IP address, hop count, IP of the predecessor on the path to the destination, IP of the next hop on the path to the destination, time-to-live value, protocol mode, action mode*). Also important to mention is the use of a distance table in these modes, as in WRP, which contains for every destination D and every neighbor N, the distance to D via N and the second-to-last hop (predecessor) through which this route is realized.

4.3 Neighbor detection

When a new node is detected, based on the mode of the detecting and detected nodes, some specific actions need to be taken (e.g., provide the new neighbor with routing information) in order for the adaptive protocol to function. Table 2 provides an overview of these actions.

4.4 Handling of mode switches

When the network conditions in the ad hoc network gradually change, nodes can detect this and can adapt their mode of operation to the new network context. With network conditions, we mean all conditions that, together, form the network context such as the mobility of the nodes, the amount of traffic generated by the nodes, the applications running in the network, the specific role nodes fulfill in the network (e.g., an important server, a client or only a forwarding node), density of the network... All these conditions will impact the performance of routing protocols and can be used as input for ad hoc network simulations.

Adapting the routing behavior to the network context has to happen with minimal impact on the ongoing network traffic. This is a challenging aspect, since changing from one mode to another has its implications, not only for the node that changes its mode, but also for all the other nodes for which this mode change involves updates in their routing table. This means that in order to keep the routing tables consistent and to maintain mode compatibility, appropriate actions need to be taken. As all mode switches can occur, the protocol should be able to deal with them. This section describes the desired protocol behavior upon mode switches. This behavior has been verified through simulations in GloMoSim and guarantees that during mode switches connectivity between applications is maintained. As all mode switches involve transitions from or to a proactive mode, between proactive modes or changes within a proactive mode, we have chosen to manage these mode switches through the propagation of special proactive update entries.

For all possible mode switches, we will give an overview of the actions that need to be taken in order to

Table 1 Mode impact on routing table

Routing table entries node *X* has when communicating with nodes *X*1, *X*2, and *X*3 (*X* = A, B, or C)			
	MODE 1 entry	**MODE 2 entry**	**MODE 3 entry**
(A, MODE 1)	A1	-	A2
(B, MODE 2)	B1	B2	B3 (= C3)
(C, MODE 3)	C1	C2	C3 (= B3)

Table 2 Actions upon detection of a new neighbor

Mode detecting node A	Mode detected node B	Action by node A (apart from increasing sequence number)
MODE 1	MODE 1	Send proactive MODE 3 entries with TTL > 0
MODE 2	MODE 1	Send proactive MODE 3 entries with TTL > 0
MODE 3	MODE 1	Send proactive MODE 3 entries with TTL > 0
MODE 1	MODE 2	Send proactive MODE 3 entries with TTL > 0
MODE 2	MODE 2	Send proactive MODE 2 and MODE 3 entries with TTL > 0 (including own entry)
MODE 3	MODE 2	Send proactive MODE 2 and MODE 3 entries with TTL > 0 (including own entry)
MODE 1	MODE 3	Send proactive MODE 3 entries with TTL > 0
MODE 2	MODE 3	Send proactive MODE 2 and MODE 3 entries with TTL > 0 (including own entry)
MODE 3	MODE 3	Send proactive MODE 2 and MODE 3 entries with TTL > 0 (including own entry)

keep the routing tables consistent and to maintain compatibility. The verification of these actions through simulations has been left out. For some mode switches, multiple solutions that differ both in complexity and in the number of messages involved are sometimes possible. The most intuitive and straightforward solution that incorporates the least overhead has been selected and described. We assume all nodes are minimally capable of supporting MODEs 1 and 3, although in reality the support of MODE 3 could be seen as optional.

4.4.1 M1_M3

When a node *N* switches from the reactive MODE 1 to the proactive MODE 3 with time-to-live value TTL, this node will first update its own proactive routing table entry (this update will happen for all mode switches and will not be mentioned explicitly anymore). Next, the node has to inform all its neighboring nodes in a region of TTL hops by sending the following update entry: (*N*, 0, *N, N, TTL* - 1, *MODE 3, M1_M3*). On reception of the update, the receiving nodes will create an entry for the new destination in their distance table and proactive routing table and will update their main routing table. Then, the nodes will propagate the update further in the network like a regular proactive update entry (ActionMode = DEFAULT), thereby decrementing its time-to-live value and updating the hop count and next hop information (the directly neighboring nodes also have to replace the predecessor address with their own address). Once the time-to-live value has become zero, propagation of the update in the network stops. When a directly neighboring node that is in MODE 2 or MODE 3 receives the update, the following actions need to be taken. As now the switching *n* ode is also allowed to process MODE 2 entries, the directly neighboring nodes that are in MODE 2 or MODE 3 need to send their proactive MODE 2 entries that have a time-to-live value greater than zero. Figure 3 and Table 3 give an overview of the update entries sent by the nodes assuming a time-to-live value of 3 (the table does not include the MODE 2 entries nodes A and C have to send upon reception of the update).

4.4.2 M1_M2

In case a node switches from MODE 1 to MODE 2 the same actions will be taken as in the case of a mode switch from MODE 1 to MODE 3, except for two differences.

First, only nodes in MODE 2 or in MODE 3 are allowed to process and further propagate the MODE 2 routing information. Second, the update entry sent by node *N* will have MODE 2 as the value for the mode field and M1_M2 as the value for the ActionMode field.

4.4.3 M3_M1

As already stated, a node in MODE 1 can only have MODE 3 routing table entries in its proactive routing table. This means that in case a node switches from MODE 3 to MODE 1, it has to remove all MODE 2 routing table entries from the proactive routing table. If no measures are taken, all active connections that use these entries will lose their route to the destination. In order to avoid this behavior, a connection table keeps track of all on-going connections. Whenever a node switches to MODE 1, the MODE 2 entries for active connections are replaced by an entry in the reactive routing table. This change will have no impact on on-going communication. As shown in Figure 4, node *N* is in MODE 3 and has proactive routing table entries for nodes A and C, both in MODE 2, and nodes B and D, both in MODE 3. There is one active connection between nodes A and B. When node *N* switches from MODE 3 to MODE 1, *N* has to remove all its MODE 2 routing table entries and has to convert those that are part of an active connection to MODE 1 entries.

The end result is a proactive routing table that only contains MODE 3 entries and a reactive routing table

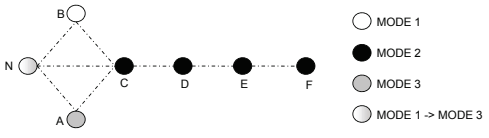

Figure 3 Node N switches from MODE 1 to MODE 3.

Table 3 Update entries sent when *N* switches from MODE 1 to 3

Node sending update	Update entry
N	(*N*, 0, *N*, *N*, 2, MODE 3, M1_M3)
A	(*N*, 1, A, *N*, 1, MODE 3, DEFAULT)
B	(*N*, 1, B, *N*, 1, MODE 3, DEFAULT)
C	(*N*, 1, C, *N*, 1, MODE 3, DEFAULT)
D	(*N*, 2, C, C, 0, MODE 3, DEFAULT)
E	TTL expired
F	TTL expired

that contains an entry to node A that was previously proactive, but that has been converted to a reactive entry. It is clear that all packets from the active connection, which are either destined for node A or node B, can still be routed to the destination. As such, the mode switch has no impact on the on-going communication.

There is still one issue that needs to be resolved. When the switching node removes or replaces its MODE 2 routing table entries, other nodes that have a MODE 2 entry for these destinations that use the switching node as next hop are also unable to use the proactive path through the switching node, as this path will not be updated anymore. For instance, in Figure 4, node A will have a proactive MODE 2 routing table entry to node C, that uses node *N* as next hop. If node *N* does not participate in the proactive MODE 2 routing process anymore, the entry to node C will not be updated anymore by node *N* and the neighboring nodes have to become aware of this. Therefore, for all MODE 2 entries that have been deleted or converted by the switching node and that have a time-to-live value greater than zero, a DEGRADE_DESTINATION update entry is sent, which announces that the path to that destination through the switching node has become degraded, i.e., will not be updated anymore by the node that sent this update entry. As a consequence, the neighboring nodes that receive this update will do the following:

• They will first delete their distance table entry to that destination via the node that sent the DEGRADE_-DESTINATION update.

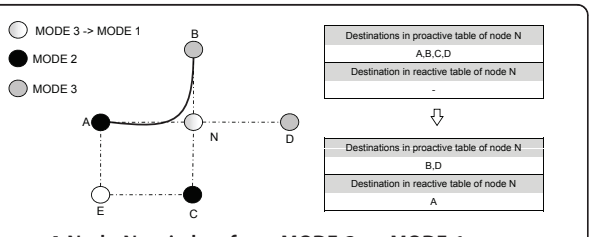

Figure 4 Node N switches from MODE 3 to MODE 1.

• If there is an active connection to that destination that uses the sender of the DEGRADE_DESTINATION update as next hop, the current proactive routing table entry is replaced by a reactive routing entry.

• Next, the proactive routing table for that destination is updated. If an alternative path is found (for instance, in the example node A would have an alternative path to node C if node E were in MODE 2 or MODE 3), this new path is added to the main routing table and announced through an update message. Note that in this case it is possible of having two routes to the same destination at the same time (the converted path and the newly calculated proactive path). The best route can then be chosen. If no alternative path is found and the time-to-live value of the old path was bigger than zero, the DEGRADE_DESTINATION update entry is further propagated.

Next, when a node is in MODE 3 with time-to-live value TTL, all nodes within a region of TTL hops will have a proactive routing table entry to this node. This implies that if this node switches to MODE 1, all these entries also need to be deleted or replaced by a MODE 1 entry, according to the above principles, if the MODE 3 routing table entry is being used by an active connection. To this end, the node that switches from MODE 3 to MODE 1 will send the following update message: (*N*, 0, *N*, *N*, -1, *MODE 1, M3_M1*).

When a node receives the update entry, it will first check if it still has a MODE 3 entry to that destination in its proactive routing table. If not, the update entry is discarded. If there is a MODE 3 entry for that destination, the node receives the M3_M1 update entry for the first time. If the path to the destination is in use by an active connection, the entry is converted to a MODE 1 entry, based on the information in the proactive routing table. Next, the node will invalidate all MODE 3 entries for that destination in its distance table. If no valid entries exist anymore, then the destination is removed from both the distance table and routing table, else only the proactive routing table entry is invalidated. Next, the MODE 3 entry for that destination is removed from the main routing table. Finally, if the time-to-live value of the removed MODE 3 entry was greater than zero, then the M3_M1 update entry is further forwarded to all neighboring nodes, after updating the necessary fields (i.e., with predecessor, next hop... information according to the removed MODE 3 entry).

4.4.4 M2_M1

In case a node switches from MODE 2 to MODE 1, almost the same actions will be taken as in the case of a mode switch from MODE 3 to MODE 1. First, the node will convert all its MODE 2 entries that are in use to MODE 1 entries. The other MODE 2 entries will be deleted. Also, for all MODE 2 entries that have been

deleted or converted by the switching node and that have a time-to-live value greater than zero, a DEGRADE_DESTINATION update entry is sent, which is processed in the same way as discussed in the previous section. Finally, the node that switches from MODE 2 to MODE 1 will also send the following update message: (*N*, 0, *N*, *N*, -1, *MODE 1*, *M2_M1*). However, in this case this update will only be processed by and forwarded between MODE 2 and MODE 3 nodes, as these are the only ones that can have a MODE 2 entry in their routing table. All other actions are similar to the ones discussed in the previous section.

4.4.5 M2_M2_TTL_INC, M2_M2_TTL_DEC, M3_M3_TTL_INC, and M3_M3_TTL_DEC

When a node stays in the same proactive mode (e.g., MODE 2 or MODE 3), but increases or decreases its time-to-live value in order to change the scope where its updates are propagated, this information needs to be forwarded to all relevant nodes. In case of an increase of the time-to-live value of a node, this information needs to be propagated to all nodes that already have an entry to that node, but also to all new nodes that now fall within the new, larger scope of propagation. To this end, updates that only announce a change in time-to-live value of existing paths will also be further propagated in the network.

In case of a decrease of the time-to-live value, the information also needs to be propagated to all nodes that have an entry to the node that changed its time-to-live value (i.e., all nodes within the scope determined by the old time-to-live value). All of these nodes that are still within the scope determined by the new time-to-live value will update the time-to-live information. All nodes that are not within this scope anymore due to the new, but lower, time-to-live value, have to remove the entry from their routing tables, replacing it by a reactive entry if the path was in use by a connection. To this end, an update with the new TTL value will be sent by the switching node and will be propagated. When the time-to-live value in the update becomes zero, the receiving node is not allowed to further propagate the update. However, as the old time-to-live value was higher than zero, the neighboring nodes need to be informed about this. As we have already discussed, this can be done through the propagation of a DEGRADE_DESTINATION update entry. Upon reception of this DEGRADE_DESTINATION update entry, the same actions will be taken as before. Figure 5, and Tables 4 and 5 illustrate the behavior of the protocol for these 2 mode switches. The same actions will be taken when a node in MODE 3 changes its time-to-live value, except for the fact that the propagated updates will also be processed by MODE 1 nodes.

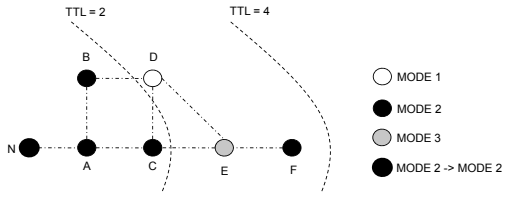

Figure 5 Node N in MODE 2 changes its time-to-live value.

4.4.6 M2_M3 and M2_M3_TTL_DEC

The next mode switch we will discuss is the switch of a node from MODE 2 to MODE 3, where the time-to-live value can increase, stay the same, or decrease. First, the node that switches from MODE 2 to MODE 3 only has to perform two actions: the node will first update its proactive routing table entry and then, it has to inform all neighboring nodes of its mode switch. For informing its neighbors, the node that switches its mode will simply send a DEFAULT update entry that includes the new mode and its time-to-live value to all its neighboring nodes. Upon reception of this update by neighboring nodes, these nodes will update their distance table. The update will now also be propagated if only a change in mode took place. As such, updates that only announce a change in mode of existing paths will also be further propagated in the network. All other actions remain the same. Figure 6 and Table 6 show an example of this mode switch and the corresponding update entries sent. Also note that in this example the switch leads to the creation of a shorter path to node *N*.

4.4.7 M3_M2 and M3_M2_TTL_DEC

The last mode switch we will discuss is the switch of a node from MODE 3 to MODE 2. Again, the time-to-live value can increase, stay the same, or decrease. The main difficulty to handle this mode switch is the following. Every node in MODE 1 that is made aware of the mode switch has to remove the existing MODE 3 entry for the reasons explained above. However, when a node in MODE 2 is made aware of the mode switch the actions that need to be taken depend on the specific topology of the network.

Table 4 Update entries sent when node *N* in MODE 2 increases its time-to-live value from 2 to 4

Node sending update	Update entry
N	(*N*, 0, *N*, *N*, 3, MODE 2, DEFAULT)
A	(*N*, 1, A, *N*, 2, MODE 2, DEFAULT)
B	(*N*, 2, A, A, 1, MODE 2, DEFAULT)
C	(*N*, 2, A, A, 1, MODE 2, DEFAULT)
D	Not allowed to participate in the process
E	(*N*, 3, A, C, 0, MODE 2, DEFAULT)
F	TTL expired

Table 5 Update entries sent when node *N* in MODE 2 decreases its time-to-live value from 2 to 4

Node sending update	Update entry
N	(*N*, 0, *N*, *N*, 1, MODE 2, DEFAULT)
A	(*N*, 1, A, *N*, 0, MODE 2, DEFAULT)
B	(*N*, INF, INV, INV, -1, MODE2, DEG_DST)
C	(*N*, INF, INV, INV, -1, MODE2, DEG_DST)
D	Not allowed to participate in the process
E	(*N*, INF, INV, INV, -1, MODE2, DEG_DST)
F	Old TTL expired

Consider the two network topologies given in Figure 7. In the upper topology, when node *N* switches from MODE 3 to MODE 2, node B has to remove the existing MODE 3 entry to node *N*. The same holds for nodes E, F, and G. However, in the lower topology, the situation is completely different. Here, node B does not have to remove the proactive entry to node *N*, as an alternative path is available through node D. The same holds for node H. However, nodes E, F, and G still need to be informed of the fact that it has to remove its existing MODE 3 entry. It is clear that this creates an ambiguous situation.

In order to solve this problem, we propose to announce the mode switch as the concatenation of two switches we have discussed previously, namely a switch from MODE 3 to MODE 1 followed by a switch from MODE 1 to MODE 2. This means that when a node switches from MODE 3 to MODE 2 it has to send two update entries, contained in the same update message.

However, the following remark has to be made. Assume the switching node sends both M3_M1 and M1_M2 update entries, in that order. When the M1_M2 update is processed by a neighboring node that is in MODE 2 or MODE 3, this node will send its MODE 2 routing table entries with time-to-live higher than zero to the switching node. This action is not needed, as the switching node its previous mode was MODE 3, which means it was already allowed to process these MODE 2 entries. To this end, the update entry sent will contain M1_M2_l as ActionMode value, instead of M1_M2, indicating that the neighboring nodes in MODEs 2 and 3 do not have to send their MODE 2 entries with time-to-live value greater than zero.

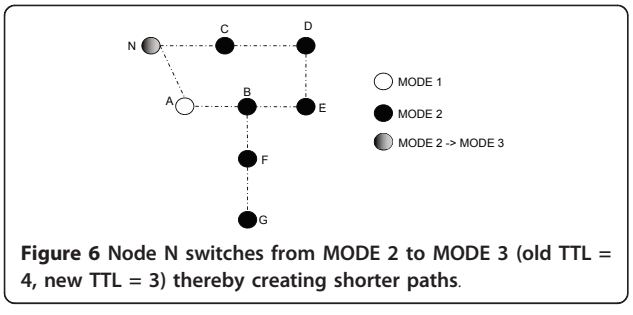

Figure 6 Node N switches from MODE 2 to MODE 3 (old TTL = 4, new TTL = 3) thereby creating shorter paths.

Table 6 Update entries sent when node *N* switches from MODE 2 to MODE 3 (old TTL = 4, new TTL = 3)

Node sending update	Update entry
N	(*N*, 0, *N*, *N*, 2, MODE 3, DEFAULT)
A	(*N*, 1, A, *N*, 1, MODE 3, DEFAULT)
C	(*N*, 1, C, *N*, 1, MODE 3, DEFAULT)
B	(*N*, 2, A, A, 0, MODE 3, DEFAULT)
D	(*N*, 2, C, C, 0, MODE 3, DEFAULT)
E	(*N*, INF, INV, INV, -1, MODE 3, DEG_DST)
F	New TTL expired
G	-

All other actions for the processing of the update entries are the same as discussed previously and this is illustrated in Figure 8 and Table 7.

4.5 Link break detection and handling

Due to mobility or changes in the signal quality, the link between two neighboring nodes can break, resulting in the unavailability of routes that use that link. The routing protocol should be able to react to these changes in the network topology by establishing new routes.

First, when a node detects a link break to a neighboring node, the node will remove this neighbor from its neighbor table and distance table. Then, the node has to check which routes are affected by the link break. If there are MODE 1 entries in its routing table that use the neighboring node as next hop, then a MODE 1 route error message is broadcasted that announces all destinations that have become unreachable via the node that broadcasts the route error. If there is a MODE 2 or MODE 3 entry for the neighboring node, the node that detects the link break will also send a DEFAULT update entry that announces that the link has become unavailable. For all other MODEs 2 and 3 entries (thus for destinations different

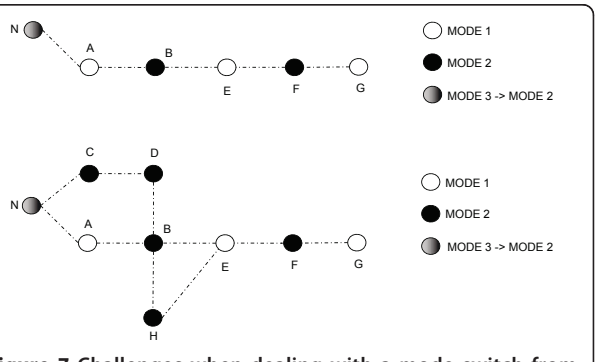

Figure 7 Challenges when dealing with a mode switch from MODE 3 to MODE 2.

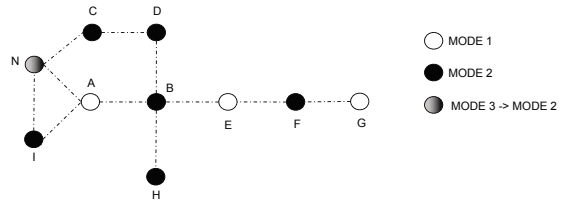

Figure 8 Node N switches from MODE 3 to MODE 2 with both the old and new time-to-live value equal to 4.

from the neighboring node) that are using the neighboring node as next hop, the proactive routing table is updated. When no alternative path is found, a DEFAULT update entry that announces that the destination has become unreachable is broadcasted. If an alternative path exists and the time-to-live value is greater than zero, a DEFAULT update entry that announces the new path is sent. On the other hand, if an alternative path exists, but its new time-to-live value is zero and its old time-to-live value was greater than zero, the node will send a DEGRADE_DESTINATION update entry, to announce to its neighboring nodes that it will not send any further proactive updates for that destination. In this way, the time-to-live values and the propagation of the updates based on these values are kept consistent. Figure 9 and Table 8 show the actions taken upon a link break.

5. Evaluation

In the previous section, we have only evaluated the actions that need to be taken upon a single mode switch and we have verified through simulations that these actions resulted in a consistent state of the routing tables. In order to be really able to exploit the potential of the adaptive routing protocol, similar behavior needs to be obtained when multiple nodes are switching their mode of operation or nodes in a certain mode are moving within the network. To this end, we have simulated the behavior of the adaptive routing protocol in two scenarios, which will now be discussed, and compared it to traditional proactive and reactive routing in order to have a reference point. These simulations illustrate that, thanks to the realized mode compatibility, our protocol allows different nodes to have different modes and to move freely around and to individually switch mode, without service discontinuity, without impacting ongoing connections, while continuing to forward traffic to its destination. As such, these simulations have to be seen as a verification of the algorithmic behavior and correct implementation of the protocol. Even in scenarios with randomly assigned initial modes, mobile nodes or nodes randomly switching their mode, reasonable performance can be achieved. Of course, only by applying the mode switches intelligently, increased performance compared to reactive and proactive protocols can be achieved, which is subject of Section 6.

5.1 Static network with random mode switches

In a static network of 36 nodes that are placed in a grid topology with grid space equal to 175 m and that each have one 802.11b radio with a send range of 250 m at 11 Mbit/s, the performance (packet delivery ratio, end-to-end delay, and the number of routing control packets) of pure proactive and reactive routing is evaluated for two different traffic patterns (10 and 20 traffic

Table 7 Update entries sent when node *N* switches from MODE 3 to MODE 2 (old TTL = new TTL = 4)

Node sending the update	Update entry
N	(*N*, 0, *N*, *N*, -1, MODE 1, M3_M1)
	(*N*, 0, *N*, *N*, 3, MODE 2, M1_M2_I)
A	(*N*, 1, A, *N*, -1, MODE 1, M3_M1)
	Not allowed to process M1_M2_I update entry
C	(*N*, 1, C, *N*, -1, MODE 1, M3_M1)
	(*N*, 1, C, *N*, 2, MODE 2, DEFAULT)
I	(*N*, 1, I, *N*, -1, MODE 1, M3_M1)
	(*N*, 1, I, *N*, 2, MODE 2, DEFAULT)
B	(*N*, 2, A, A, -1, MODE 1, M3_M1)
	(*N*, 3, C, D, 0, MODE 2, DEFAULT)
D	(*N*, 2, C, C, -1, MODE 1, M3_M1)
	(*N*, 2, C, C, 1, MODE 2, DEFAULT)
E	(*N*, 3, A, B, -1, MODE 1, M3_M1)
H	(*N*, 3, A, B, -1, MODE 1, M3_M1)
	TTL of DEFAULT update entry expired
F	Old TTL expired, M3_M1 update not forwarded anymore
G	-

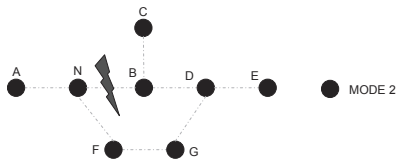

Figure 9 Node N detects a link break to node B. Nodes B, C, D and E all have time-to-live value 4.

sources with a constant bit rate UDP traffic at a rate of 112 kbit/s), each averaged over 10 runs. Next, using the same traffic patterns, the simulations are redone with our adaptive routing protocol. At the beginning of the simulation, the routing modes are randomly assigned (MODE1, MODE2 - TTL 8, or MODE3 - TTL 8). During the simulation, X random mode switches per minute occur, with X ranging from 1 to 256 and X being the total number of switches of all nodes. All mode switches are averaged over five runs, each resulting in a different initial assignment of the modes and the mode switches that occur during the simulation. Together with the different runs for the traffic patterns, the presented results are averaged over 25 runs.

Figure 10 shows the results. When looking at the results for ten simultaneous traffic connections (connections change approximately every 100 s), we see that purely proactive routing clearly outperforms reactive routing in terms of packet delivery ratio and end-to-end delay, but exhibits a higher control overhead. When using our adaptive routing protocol where the initial modes are randomly assigned to nodes, we can observe that for low mode switch frequencies (i.e., frequencies lower than 4 switches per minute), the performance is in between the performance of reactive and proactive routing. In this setting, the network is a random mixture of proactive and reactive routing and our protocol is able to correctly deliver most packets to the destination and to achieve a performance as expected, i.e., somewhere between the performance of proactive and reactive routing. When the mode switch frequency increases, the packet delivery ratio gradually decreases (Figure 10a), the end-to-end delay the data packets experience increases (Figure 10b) and there is an increasing amount of routing control packets (Figure 10c). This is as expected, since every additional random mode switch introduces additional instability in the network, resulting in new routing updates, rerouting of traffic, additional congestion... It is also interesting to observe that the deviation of the results–except for the amount of routing control packets since this amount mainly depends on the number of mode switches–strongly increases when the mode switch frequency increases. Every random mode switch adds additional uncertainty to the simulation (which node, which mode switch, mode of neighboring nodes...), increasing the potential simulation space. Even averaged over 25 runs, the deviation of the results can therefore become quite high. It also confirms once more the strong impact of the mode switch randomness on the results.

Table 8 Update entries sent by node *N* upon detection of the link break to node B

Update entry
(B, INFINITY, INVALID, INVALID, -1, MODE 2, DEG_DST)
(C, INFINITY, INVALID, INVALID, -1, MODE 2, DEFAULT)
(D, 3, G, F, 0, MODE 2, DEFAULT)
(E, INFINITY, INVALID, INVALID, -1, MODE 2, DEG_DST)

Since random mode switches have been used and mode switching is not triggered by network changes, these results say nothing about the actual performance gain that can be achieved when the mode switching capabilities are applied intelligently according to the network context. With random mode switches or badly assigned modes it can even occur that suboptimal routes in terms of hop count are being used (e.g., when routing via a long U-shaped proactive zone takes place, while a shorter non-proactive route could exist). Also very important to remark is that with intelligent mode switching behavior, nodes will only switch when needed and will possibly switch in group, since groups of nodes will often experience a similar network context. This is important, because when the mode switch frequency becomes too high, the packet delivery ratio drops. This is caused by the excessive amount of additional routing control packets (Figure 10c) that is generated in case of very high mode switch frequencies increasing the convergence time of the protocol.

Consequently, adaptivity is only beneficial when adapting to medium and large time-scale network changes, i.e., changes in network context that are observed during periods in the order of tens of seconds (or multiple minutes), since these will only trigger a limited number of mode switches. Short time-scale events, e.g., overflow in a queue, short period of congestion, bad link..., should therefore not trigger a mode switch, but should be handled by other mechanism that are able to act at a lower time scale such as TCP congestion control, efficient scheduling, link adaptation... As such a combination of short time-scale mechanisms in combination with adaptive routing to handle larger time-scale changes could be an appropriate fully adaptive solution. Of course, multiple adaptive mechanisms operating at different time scales and in different layers can affect each other and would constitute a complex research problem on its own.

Figure 10d-f, where the traffic load is increased, also confirms the above observations. Now, with the

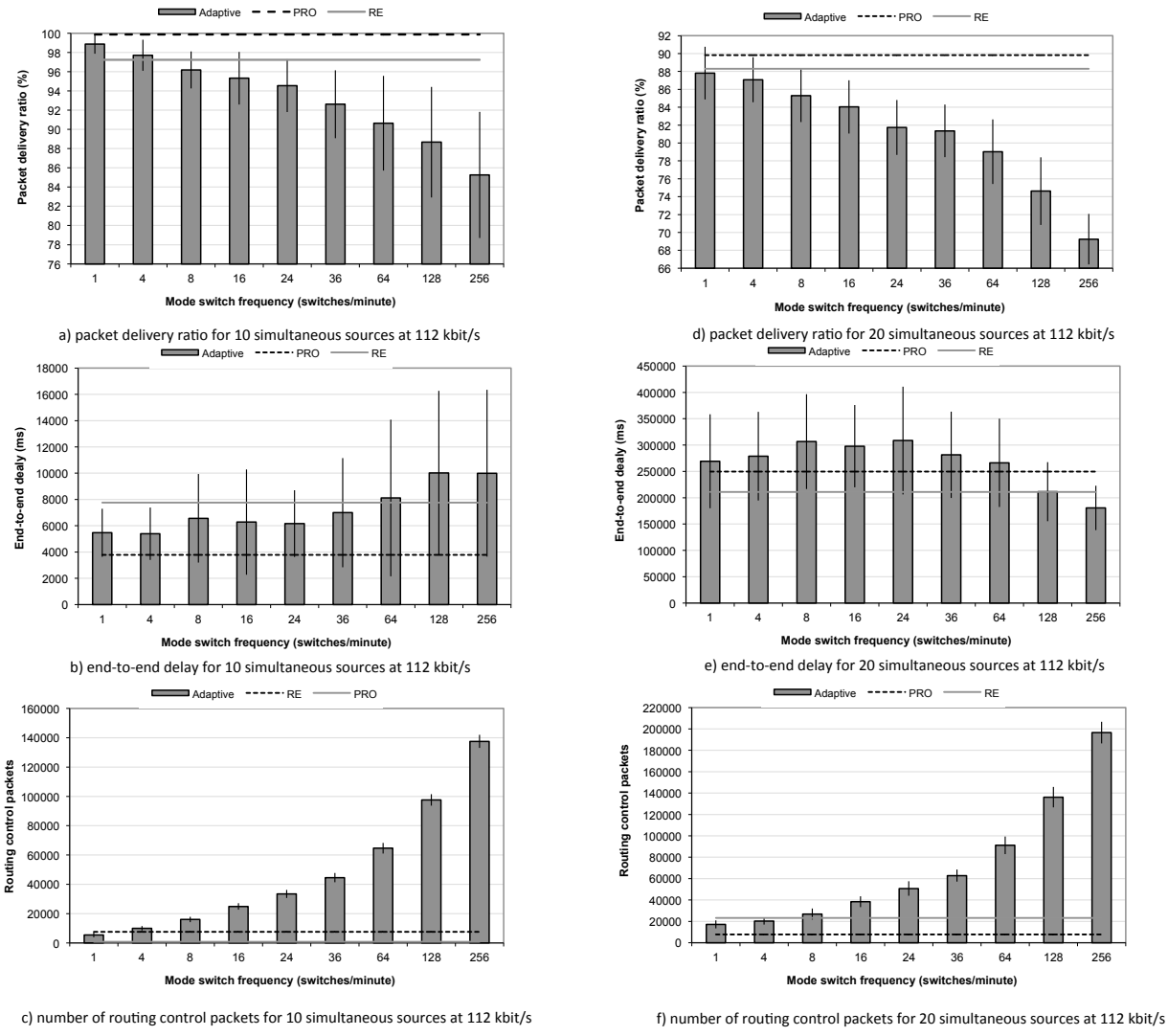

Figure 10 Static network with random mode switches.

increased traffic load, it becomes even more difficult to achieve good performance when the mode switch frequency, and thus the number of routing control packets, becomes too high. A low number of random mode switches again already leads to acceptable results, which could be greatly improved when exploiting the adaptive capabilities intelligently. Consequently, from these simulations it can be concluded that, in a static network, the adaptive routing protocol can handle random individual mode switches while maintaining reasonable or sometimes similar performance than proactive and reactive protocols. As such, it has the potential to outperform traditional ad hoc routing protocols (or at least perform as well) provided mode switches are triggered intelligently according to the network context, and that they are triggered for handling medium to large time-scale network context changes in order to avoid excessive control traffic and a resulting performance degradation.

Note that we did not evaluate the event of all nodes switching their mode simultaneously. In this case, the result would strongly depend on the exact mode switch that occurs. If all nodes would switch to the same mode (e.g., from reactive MODE1 to proactive MODE2 with the same TTL), performance would be comparable to bootstrapping a 36-node proactive network simultaneously. If the mode switches were random, result would be in line with the results in Section 5, i.e., the performance would temporarily degrade and control traffic would temporarily increase.

5.2 Mobile network with fixed modes

In the previous section, we considered mode switching in a static network. Now we will investigate the behavior

of nodes with different pre-assigned random modes in a mobile network and compare it with proactive and reactive routing. Again a network of 36 nodes is used. This time, the nodes are distributed randomly over an area of 750 m × 750 m and move according to the random waypoint model with a fixed speed that ranges from 2 to 10 m/s and a pause time equal to 0 s. Modes are assigned randomly to the nodes according to ten different distributions shown in Table 9 and remain the same for the duration of the simulation. As such, the behavior and performance of the protocol in the presence of mobility is evaluated. The TTL value is set to 16, in order to also have a completely proactive network. Every moment ten traffic sources are active simultaneously. All other parameters are the same as in the previous experiments. Again, we have used 5 different runs for the traffic and for every run, 5 different assignments of the modes and different seeds are used, leading to results that have been averaged over 25 measurements.

The results are shown in Figure 11. When looking at the packet delivery ratio, it is clear that almost for all mode distributions quite similar packet delivery ratios are achieved. This is a consequence of the fact that for all mode distributions maintenance of the routes is needed due to mobility, causing similar packet losses.

More important to note is that this result proves that the correctness of the protocol is guaranteed when using nodes in different modes in the presence of mobility. When nodes are mobile, new links and link breaks with nodes in similar or different modes will occur, requiring different actions of the adaptive protocol. When looking at the routing control overhead, it is clear that when the amount of proactiveness in the network increases, the number of routing control packets needed also increases. Again, no optimal mode assignment has been done, but it has been proven that nodes can individually and randomly select a mode and protocol correctness remains guaranteed also in mobile networks. This again illustrates the correctness of the protocol implementation and reveals the potential of the adaptive routing protocol when mode switches are applied intelligently.

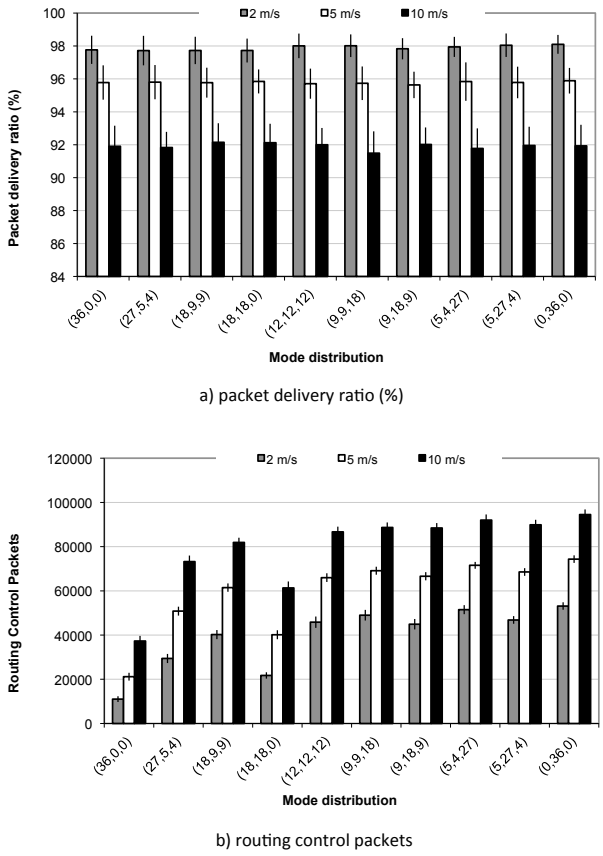

Figure 11 Mobile network with predefined mode distribution.

In the following section, we will illustrate how the adaptive behavior of the protocol can be exploited in order to achieve better results compared to traditional ad hoc routing protocols.

6. Applications

6.1 Adaptivity to network context

In this section, we will illustrate with a concrete example how the protocol can adapt to varying network

Table 9 Mode distribution over nodes

Number of nodes in MODE 1	Number of nodes in MODE 2 TTL = 12	Number of nodes in MODE 3 TTL = 12
36	0	0
0	36	0
18	18	0
12	12	12
18	9	9
9	18	9
9	9	18
27	5	4
5	27	4
5	4	27

conditions. Before starting the development of our adaptive protocol, we have extensively evaluated the performance of AODV and WRP under varying network conditions and have encountered several situations where due a changing network context one protocol could outperform the other. In order to demonstrate the potential of our adaptive protocol to switch its behavior according to the network context and outperform traditional protocols, we have selected one scenario based on the simulation results shown in Figure 12. These simulation results show that under an increasing network load WRP starts to outperform AODV.

Therefore, we have implemented a simple network monitoring agent that determines the network load based on the number of packets to route and the number of neighbors. This information is exchanged with the neighboring nodes by broadcasting hello messages in order for a node to obtain a view of the load in the network in its 1-hop environment. When the observed network load exceeds a certain threshold, which was now manually determined from our simulation results, nodes change their mode of operation from reactive to proactive. Once the load falls below this threshold, the mode is set back to reactive.

We simulated the performance of the adaptive multimode ad hoc routing protocol (AMAHR) in a 50-node static network, with nodes randomly distributed in a rectangular region of size 600 m × 600 m. Packets of size 512 bytes are sent at a rate of 10 packets per second. The number of sources is initially set to 5. After 1,300 s, the number of sources is increased to 20 and after 2,500 s the number of sources is set back to 5. The transmission range of all nodes is approximately 200 m and the MAC layer model used is 802.11b direct sequence spread spectrum at 2 Mbit/s. The hello interval is 1 s in the proactive mode and 5 s in the reactive mode. Again, the performance metrics considered are the packet delivery ratio, the end-to-end delay and the number of control packets per data packet delivered. Figure 13 shows the evolution of these three performance metrics over time.

The results clearly show that AMAHR combines the advantages of both proactive and reactive routing by its capability to adapt to the network context in a way that does not impact any ongoing connections. Initially, the observed traffic load is low and nodes set up routes reactively. Once the number of traffic sources increases to 20, the network monitoring component detects the increase in network load. The observed network load then exceeds the defined threshold and nodes switch to proactive routing.

As a consequence, under low network loads our adaptive protocol has the high packet delivery ratio and low control overhead of AODV. The number of control packets per data packet delivered is slightly higher than the reactive routing protocol, due to the periodic exchange of hello messages needed for monitoring the network environment, but optimizations such as optimized broadcasting or adaptive broadcast intervals are possible. By changing its mode from reactive to proactive when the traffic load increases, AMAHR achieves the high packet delivery ratio and low control overhead and end-to-end delay of WRP. Only at the time nodes switch to another mode, the performance is less than the optimum, as nodes need time to detect the change in network conditions.

Our simulation results clearly show the advantages of being able to adapt the routing protocol to the network context. However, the development of more complex monitoring and decision-making functionality is needed to fully exploit the protocol's capabilities, but is a huge research topic on its own. The following section will therefore elaborate a little bit more on the field of ad hoc network monitoring.

6.2 Advanced monitoring of network context

In order to be able to use network monitoring information for adapting the behavior of the adaptive routing protocol,

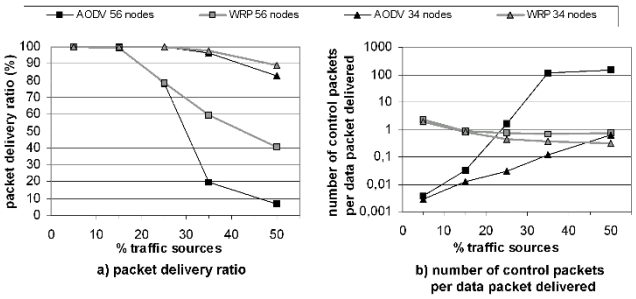

Figure 12 Performance of AODV and WRP in a 34 and 56 node static network with a density of 150 nodes/km2 under a varying percentage of traffic sources having a send rate of 10 packets per second.

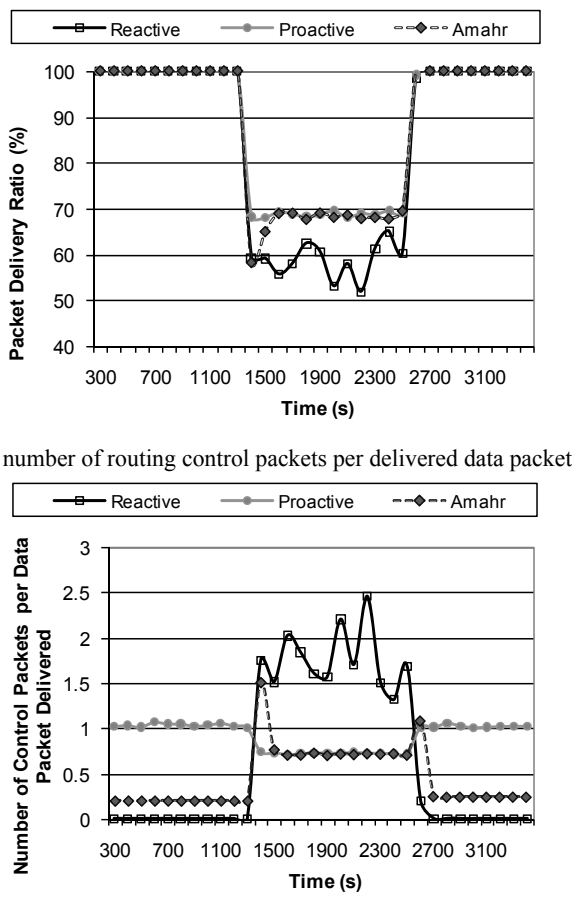

Figure 13 Performance evaluation of WRP, AODV and AMAHR in a network with changing network context.

three different requirements need to be fulfilled. First, nodes need a mechanism for monitoring network context and collecting this information. Next, some of this information needs to be propagated into the nodes' neighborhood, requiring an efficient dissemination mechanism. Finally, using the observed and averaged network context, based on collected and disseminated information, the most efficient mode should be selected possibly resulting in a mode switch. This requires a mapping between the observed network context and the different modes offered by the adaptive routing protocol, i.e., in which network context is which mode most optimal. In addition to this, learning mechanisms to analyze the impact of decisions could be added in order to make the system self-improving. Since these aspects represent a huge research topic on their own, we will only briefly discuss in this section some related work that focuses on these aspects and that could be used to complement our adaptive routing protocol developments.

DAMON [20] is a distributed architecture for monitoring mobile ad hoc networks. Agents distributed in the network collect monitoring info (energy left, traffic logs...) and send this information to their nearest sink. With this solution, only sinks will know about the network context. Since not all nodes know about the network context, they cannot individually decide to switch mode. Alternatively, the sinks could decide that the nodes in their neighborhood should switch their mode of operation, making the adaptive behavior not completely distributed anymore. In [21], a self-organized solution for monitoring and management of ad hoc networks is proposed. Instead of monitoring the whole network, only spatio-temporal connected subsets of nodes are managed, trying to capture the most interesting nodes to manage, and from this management organization overall ad hoc network performance is monitored. This study extends [22] an ad hoc network management protocol that uses clustering. The main limitation of the study is that the selection of the subsets makes use of OLSR as the underlying routing protocol and that only a subset of interesting nodes is monitored, making it difficult for nodes to individually select an optimal routing mode.

As such we favor solutions where all nodes partake in collecting statistics and can disseminate them efficiently into their neighborhood. In [23], the authors propose a Linux monitoring tool called WANMon that can be installed on all ad hoc nodes and that monitors network, power, CPU, and resource usage. Some important information such as new links, link breaks, connections... is not monitored. Also, only local monitoring information is collected, so the monitoring information should therefore be propagated periodically to neighboring nodes. As such, this interesting monitoring tool should be extended with a framework that allows the efficient dissemination of this information. Such a framework is discussed in [24]. The basic principle, which perfectly fits the adaptive routing concept, behind this work is the following: act locally considering the global network status. Local operations are lightweight, but they lack accuracy and could therefore be inefficient. To this end, the proposed Cross-Talk architecture tries to establish a global view of a number of metrics in a distributed fashion on every node in the network. The architecture consists of two views. There is the local view containing node-specific information collected from the different layers. Next to this, there is a global view that is constructed from information gathered by a data dissemination process. For the dissemination, this local information is piggybacked onto outgoing packets. Receiving nodes will use the collected samples to compute a network wide view, thereby potentially using weights (e.g., older samples receive a lower weight, samples from further away nodes could also receive a lower weight). In [24], the authors illustrate this approach for determining a global

view of the network load, which is then used to optimize the AODV route discovery process by avoiding congested routes. In [25], push and pull-based data dissemination in mobile ad hoc networks is discussed and these strategies are rated in terms of network load and overall freshness of the data and an adaptive solution is proposed. Again, the disseminated information can be used to obtain a more global view on network context parameters, but focus here is on data instead of network metrics.

Adapting to the obtained global view requires knowledge of the performance of the used modes in the adaptive routing protocol as a function of the monitored network context. Creating a mapping between the performance and the network context parameters can be done in two ways. Through extensive simulations the available modes can be evaluated for a large range of network context parameters of influence [9,26]. This is very exhaustive since endless combinations of network topologies, dynamics, and traffic patterns exist and numerous articles presenting this kind of simulation results have already been published. Alternatively, theoretical evaluations of the performance of routing protocols as a function of the network context could be used. Also here, much work has been done [27-30], but very accurate theoretical comparisons of ad hoc routing protocols remain difficult due to the numerous factors affecting the performance and the difficulties in modeling the underlying link layer. Consequently, approximations and heuristics should be used to obtain reasonable accurate mappings between the routing overhead and network context.

Also the relation with some of the ongoing work in the field of cognitive networking is worth mentioning, where networks implement cognitive processes that can perceive current network conditions, plan, decide, act on those conditions, learn from the consequences of its actions, all while following end-to-end goals. There are many similarities with the problem described in this section. Recently, work has been presented where cognitive networking techniques are being used in order to adapt routing protocol behavior [31]. Advances in this field could constitute interesting opportunities for the design of more intelligent routing protocols.

As a conclusion, we can say that interesting work on network monitoring and data dissemination has been done. The biggest challenge remaining is the mapping of the different routing protocol modes to the observed network context. In the following sections, we will show some alternative ways on how the adaptive routing capabilities can be exploited, often facilitating the mapping between the mode and the context information that is used for selecting the optimal mode.

6.3 Other potential applications

Next to using the protocol to adapt to changes in the network context, other potential applications exist. For example, in [32], we have investigated the possibility to adapt routing behavior according to the service discovery requests in the network. As such, proactive routes are maintained to popular servers, whereas less popular services can be reached through reactive routes.

Another application could be the following. The devices in a mobile ad hoc network can be very heterogeneous in terms of processing power, memory capabilities... When using the adaptive multi-mode routing protocol, the less powerful nodes could only implement one (or a few) lightweight modes (e.g., only reactive routing, pushing data to a neighbor and letting the neighbor take care of the routing...), while still capable of cooperating with more powerful nodes that support multiple modes of operations. Alternatively, when the battery level becomes too low, modes requiring a lot of power (e.g., due to a large number of message exchanges) could be disabled. The use of a modular framework and the support of multiple compatible modes would allow this type of behavior.

Next, instead of individually adapting their routing mode to the network context, a profile could be distributed to all nodes indicating how they should route, i.e., which routing mode should be used. For example, proactive routing could take place only between devices that have a trust relationship, while reactive routing is used to communicate with other devices. By updating the profile, the routing behavior can be adapted. As such, the decision to adapt is not done distributed anymore, but is taken centrally and communicated to all nodes.

Finally, instead of each different mode in the adaptive routing protocol implementing a different routing strategy, different modes could implement the same strategy, but using different power levels when sending their packets [33]. For example, there could be three proactive modes, where the first one establishes proactive routes with the lowest power-level and the third one with the highest power level. As such a number of proactive routing tables at different power levels are obtained that can be used for power-aware routing. This approach easily fits within the adaptive multi-mode routing framework.

7. Conclusion

In this article, we have presented the implementation in a network simulator of an adaptive multi-mode routing protocol offering three different compatible modes of operations. The modes are compatible, meaning that they have been designed in such a way that every node

can individually select its own optimal mode and can adapt its mode of operation at any moment while maintaining a consistent state of all routing tables and without impacting ongoing connections. This has been proven by simulations; first, possible individual mode switches, then followed by an evaluation of the protocol in a network with random mode switches and in a mobile network with pre-assigned modes. From these simulations, it has been observed that adaptivity at the routing level is only beneficial when adapting to medium and large time-scale network changes. Short time-scale changes must be handled by other mechanisms, but care should be taken that different mechanisms at different time scales and different layers do not become counterproductive. It can be concluded that this article has showcased the feasibility of designing an adaptive protocol, which has the potential to outperform traditional protocols when its adaptive capabilities are exploited intelligently, according to the network context, and that they are triggered for handling medium to large time-scale network context changes. This potential has been illustrated by a simple scenario, in which the protocol adapts itself according to the monitored network load. Although the feasibility of an adaptive routing protocol has been demonstrated, where individual nodes can change their routing behavior, it should be mentioned that during the design and implementation of the protocol we have learned that implementing compatible modes is not straightforward. When introducing new modes, many new mode switches become possible and these should all be handled in a way that does not impact ongoing traffic. This creates complex interactions and, in this study, this behavior has only evaluated through simulations. Ideally, a theoretic evaluation of the protocol correctness and the performance of individual modes according to the network context should be done in order to know when to switch. However, this is definitely not straightforward, even for existing ad hoc protocols. Also, this article has focused on the mode switching capabilities and its potential and has only provided guidelines for advanced monitoring and alternative applications of the protocol have been provided. From this study, it becomes clear that, although adaptive routing has potential and interesting applications, there are still many open issues representing huge research topics on their own. Several of these topics also appear in other domains that have recently gained increased attention from the research community, e.g., in cognitive networking or sensor networking, and remain therefore very relevant and interesting to further explore.

Competing interests

The authors declare that they have no competing interests.

Received: 15 March 2011 Accepted: 29 March 2012
Published: 29 March 2012

References

1. C-K Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, (Prentice Hall, New Jersey, 2002)
2. S Corson, J Macker, Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. RFC 2501 http://tools.ietf.org/html/rfc2501 (1999)
3. E Royer, C Toh, A review of current routing protocols for ad hoc mobile wireless networks. IEEE Personal Commun. **6**(2), 46–55 (1999). doi:10.1109/98.760423
4. D Lang, A comprehensive overview about selected ad hoc networking routing protocols. Master's thesis, Technische Universit at Munchen (2003). doi:10.1.1.103.5006
5. X Hong, K Xu, M Gerla, Scalable routing protocols for mobile ad hoc networks. IEEE Netw. **16**(4), 11–21 (2002). doi:10.1109/MNET.2002.1020231
6. SR Das, R Castaneda, J Yan, Comparative performance evaluation of routing protocols for mobile ad hoc networks, in *Proceedings 7th International Conference on Computer Communications and Networks (IC3N)*, Lafayette, LA, pp. 153–161 (1998)
7. ZJ Haas, A new routing protocol for the reconfigurable wireless networks, in *Proceedings 6th IEEE International Conference on Universal Personal Communications*, San Diego, CA, USA, pp. 562–566 (1997)
8. C-C Yanf, LP Tseng, Fisheye zone routing protocol. Comput Commun. **30**(2), 261–268 (2007). doi:10.1016/j.comcom.2006.08.014
9. V Ramasubramanian, Z Haas, EG Sirer, SHARP: a hybrid adaptive routing protocol for mobile ad hoc networks, in *Proceedings 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Annapolis, MD, USA, pp. 303–314 (2003). doi:10.1145/778415.778450
10. P Samar, MR Pearlman, ZJ Haas, Independent zone routing: an adaptive hybrid routing framework for ad hoc wireless networks. IEEE/ACM Trans Netw. **12**(4) (2004). doi:10.1109/TNET.2004.833153
11. I Chakeres, C Perkins, Dynamic MANET on-demand (DYMO) routing. http://tools.ietf.org/html/draft-ietf-manet-dymo-21
12. T Clausen, C Dearlove, P Jacquet, The optimized link state routing protocol, version 2. http://tools.ietf.org/html/draft-ietf-manet-olsrv2-11
13. T Clausen, C Dearlove, J Dean, C Adjih, Generalized MANET packet/message format. http://tools.ietf.org/html/draft-ietf-manet-packetbb-17
14. T Clausen, C Dearlove, J Dean, MANET neighborhood discovery protocol (NHDP). http://tools.ietf.org/html/draft-ietf-manet-nhdp-15
15. C Liu, J Wu, Adaptive routing in dynamic ad hoc networks, in *Proceedings IEEE Wireless Communications and Networking Conference*, Las Vegas, NV, USA, pp. 2603–2608 (2008). doi:10.1109/WCNC.2008.457
16. S Zhao, D Raychaudhuri, Policy-based adaptive routing in mobile ad hoc wireless networks, in *Proceedings IEEE Sarnoff 2006 Symposium*, Princeton, NJ, USA, pp. 1–4 (2006). doi:10.1109/SARNOF.2006.4534724
17. X Zeng, R Bagrodia, M Gerla, GloMoSim: a library for parallel simulation of large-scale wireless network, in *Proceedings 12th Workshop on Parallel and Distributed Simulations*, Banff, Alta., Canada, pp. 154–161 (1998). doi:10.1145/278008.278027
18. CE Perkins, EM Royer, Ad-hoc on-demand distance vector routing, in *Proceedings 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, USA, pp. 90–100 (1999)
19. S Murthy, JJ Garcia-Luna-Aceves, An efficient routing protocol for wireless networks. Mobile Netw Appl. **1**(2), 183–197 (1996). doi:10.1007/BF01193336
20. KN Ramachandran, EM Belding-Royer, KC Almeroth, DAMON: a distributed architecture for monitoring multi-hop mobile networks, in *Proceedings 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, Santa Clara, CA, USA, pp. 601–609 (2004). doi:10.1109/SAHCN.2004.1381963
21. R Badonnel, R State, O Festor, Self-organized monitoring in ad hoc networks. Telecommun Syst. **30**(1-3), 143–160 (2005). doi:10.1007/s11235-005-4322-3
22. W Chen, N Jain, S Singh, ANMP: ad-hoc network management protocol. IEEE J Sel Areas Commun. **17**(8), 1509–1531 (1999)
23. D Ngo, J Wu, WANMON: a resource usage monitoring tool for ad hoc wireless networks, in *Proceedings 28th Annual IEEE Conference on Local Computer Networks*, Bonn/Königswinter, Germany, pp. 738–745 (2003). doi:10.1109/LCN.2003.1243207

24. R Winter, J Schiller, N Nikaein, C Bonnet, CrossTalk: a data dissemination-based cross-layer architecture for mobile ad-hoc networks, in *Proceedings 5th Workshop on Applications and Services in Wireless Networks*, Paris, France, (2005)
25. F Bregulla, J-H Böse, K Hahn, M Scholz, Adaptive data dissemination in mobile ad-hoc networks, in *Workshop der GI-Fachgruppe "Mobilität und Mobile Informationssysteme"*, Bonn, Germany, (2005)
26. DD Perkins, HD Hughes, CB Owen, Factors affecting the performance of ad hoc networks, in *Proceedings IEEE International Conference on Communications*, New York, USA, pp. 2048–2052 (2002). doi:10.1109/ICC.2002.997208
27. L Viennot, P Jacquet, TH Clausen, Analyzing control traffic overhead in mobile ad-hoc network protocols versus mobility and data traffic activity. Wirel Netw. **10**(4), 447–455 (2004)
28. ID Aron, SKS Gupta, On the scalability of on demand routing protocols for mobile ad hoc networks–an analytical study. J Interconn Netw. **2**(1), 5–29 (2001). doi:10.1142/S0219265901000233
29. R Hekmat, *Ad-hoc Networks: Fundamental Properties and Network Topologies*, (Springer, Dordrecht, 2006)
30. M Saleem, SA Khayam, M Farooq, On performance modeling of ad hoc routing protocols. EURASIP J Wirel Commun Netw. **2010**, 1–13 (2010). 373759, doi:10.1155/2010/373759
31. Z Zhai, Y Zhang, M Song, G Chen, A reliable and adaptive AODV protocol based on cognitive routing for Ad hoc networks, in *Proceedings 12th International Conference on Advanced Communication Technology*, Gangwon-Do, Korea, pp. 1307–1310 (2010)
32. J Hoebeke, I Moerman, B Dhoedt, P Demeester, Analysis of decentralized resource and service discovery mechanisms in wireless multi-hop networks. Comput Commun. **29**(13-14), 2710–2720 (2006). doi:10.1016/j.comcom.2006.01.025
33. V Kawadia, PR Kumar, Power control and clustering in ad hoc networks, in *Proceedings 22nd Annual Joint Conference of the IEEE Computer and Communications*, Societies, San Francisco, CA, USA, pp. 459–469 (2003). doi:10.1109/INFCOM.2003.1208697

doi:10.1186/1687-1499-2012-126
Cite this article as: Hoebeke *et al.*: **Adaptive routing for mobile ad hoc networks.** *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:126.